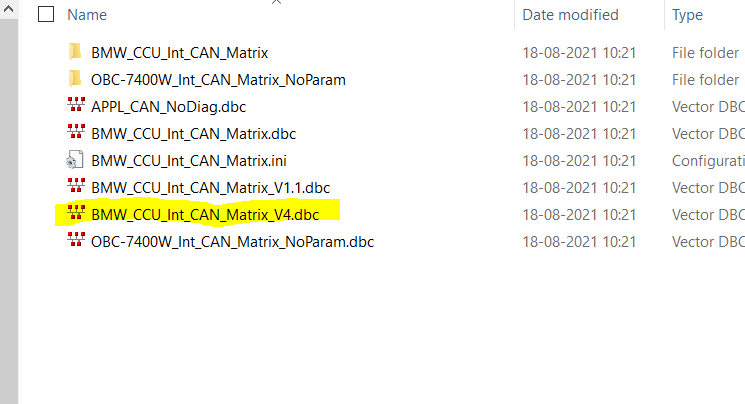
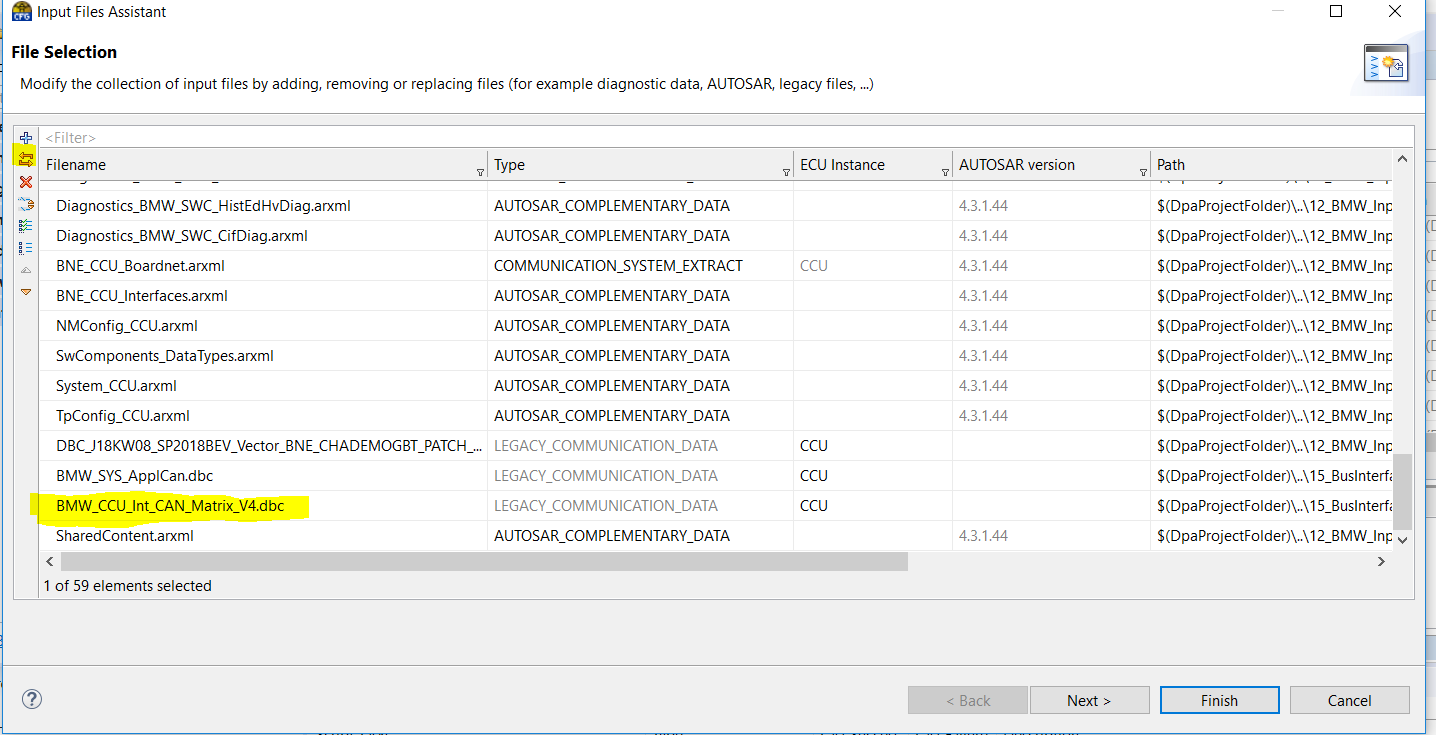
**Intcanhl**

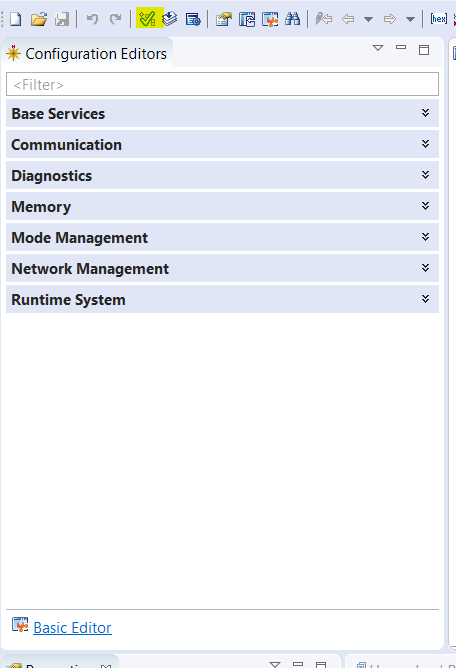
**Step1 :** Intcan DBC update :-> list down the changes of Intcan dbc from previous version.



**Step2 :** Integrate the new dbc file by replacing the previous version.

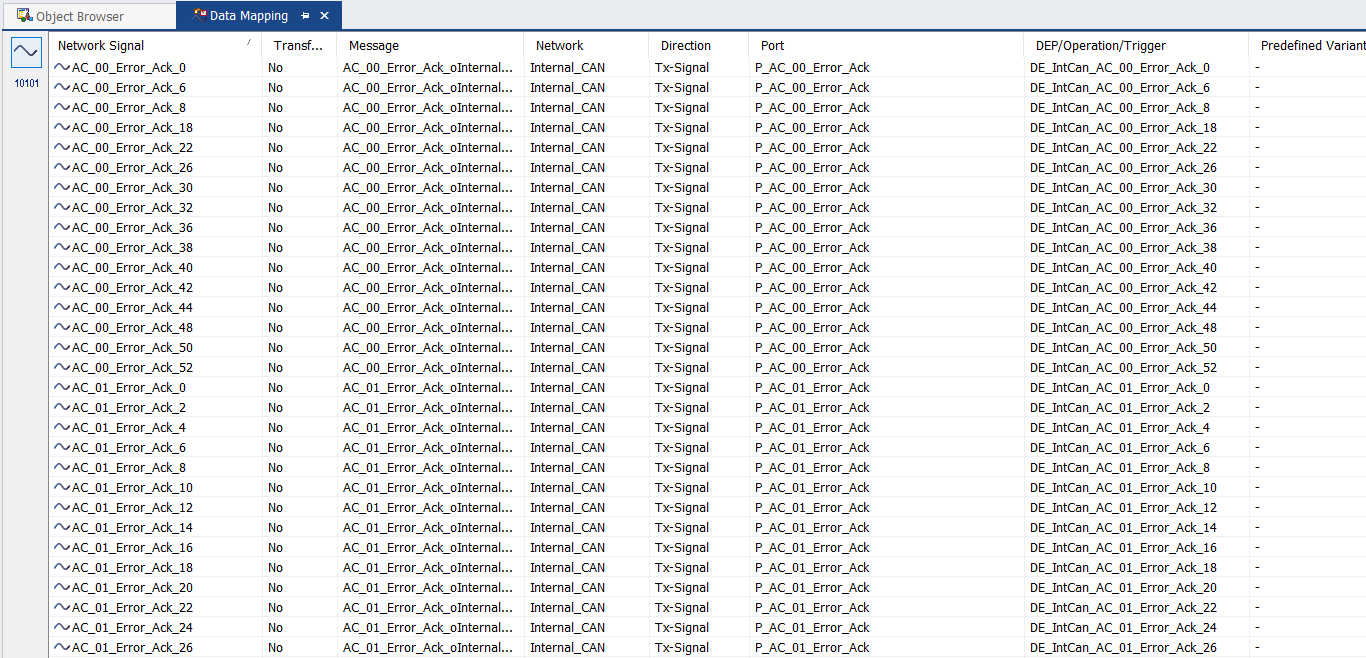


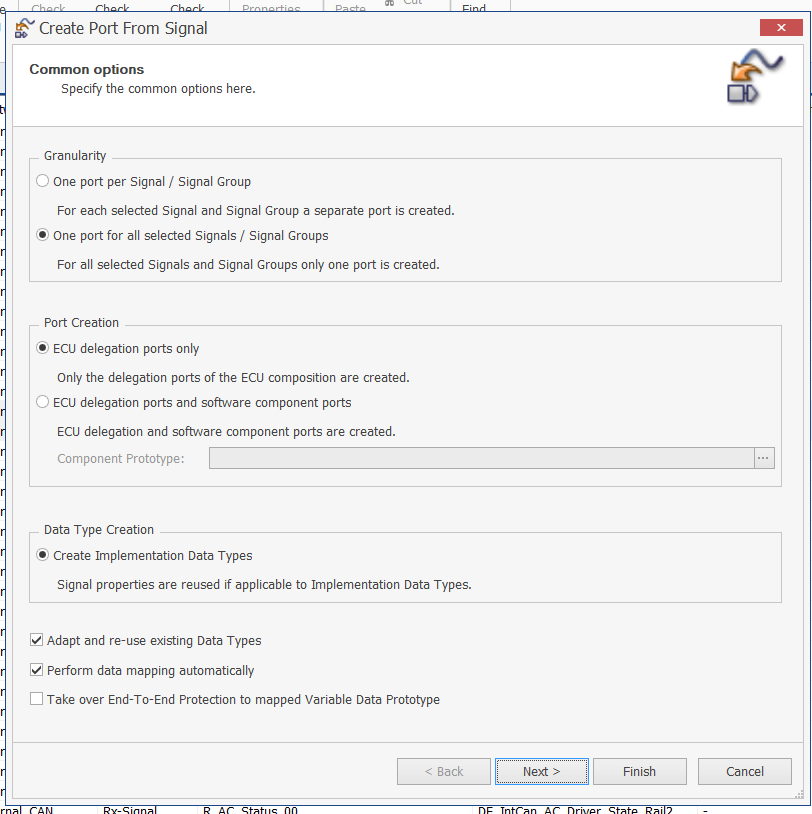
**Step3 :** Update the workflow and do on demad validation.



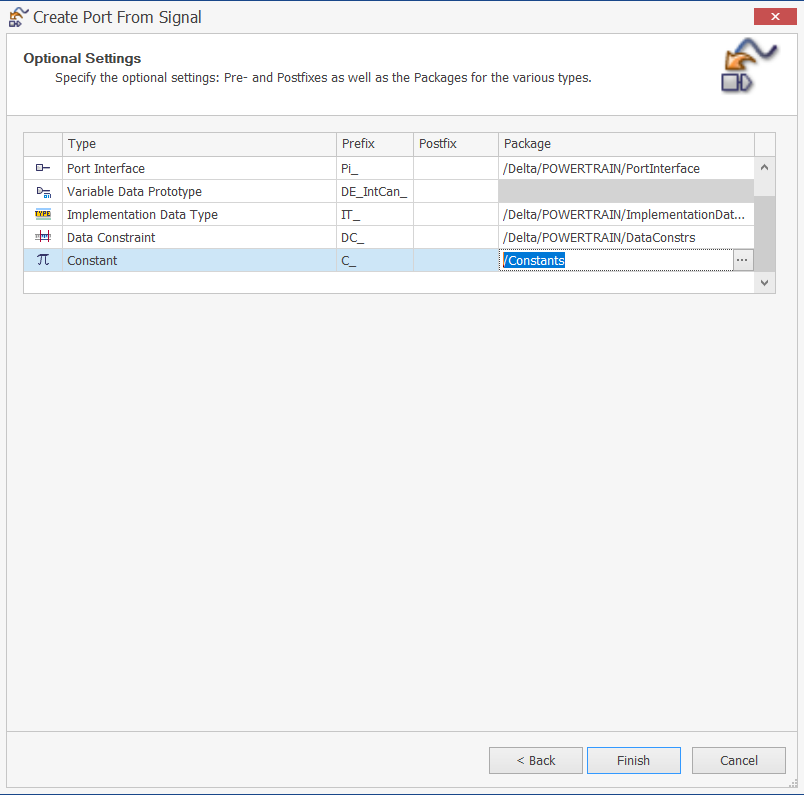
**Step4 :** Resolve the configuration error.

**Step5 :** Perform Data mapping of the Newly added signals.



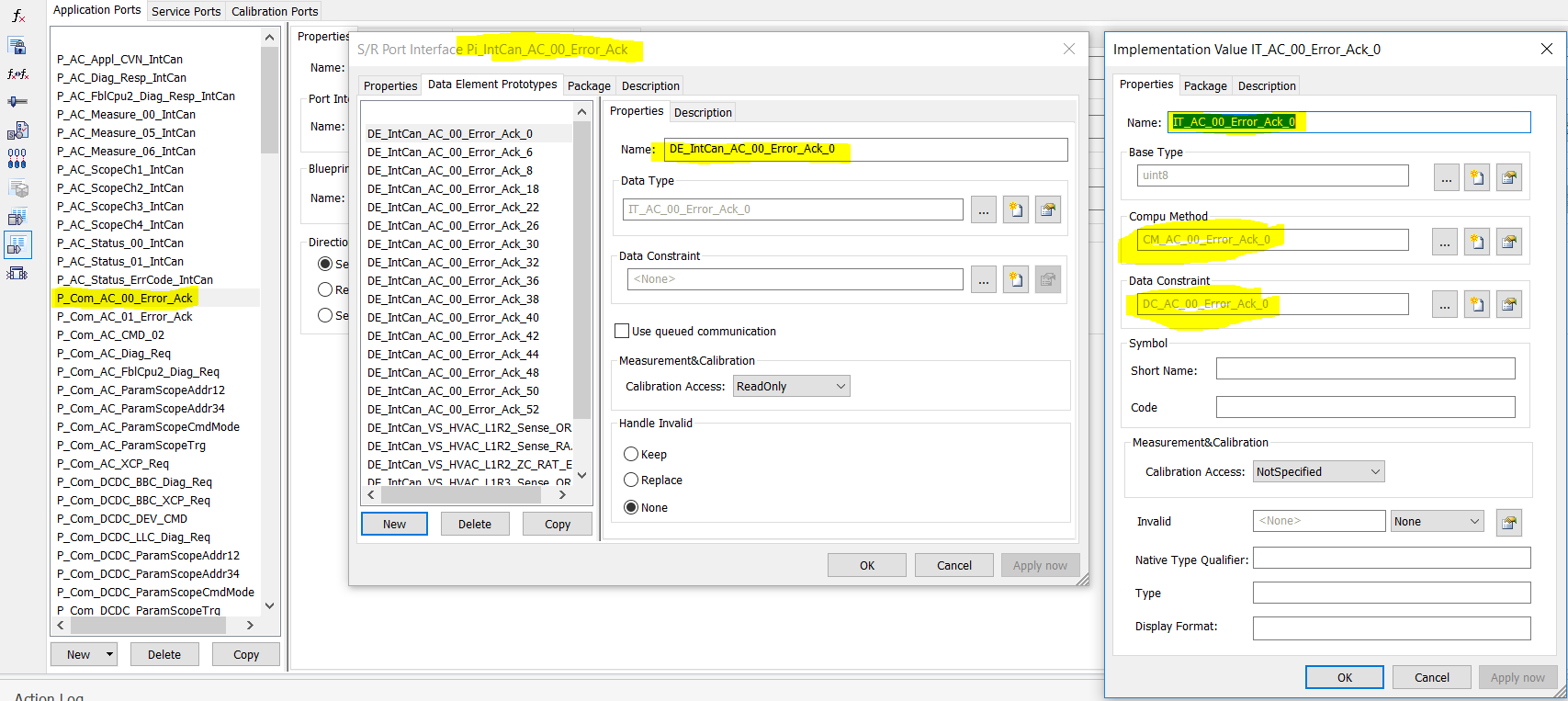


Select the correct package for creating the objects

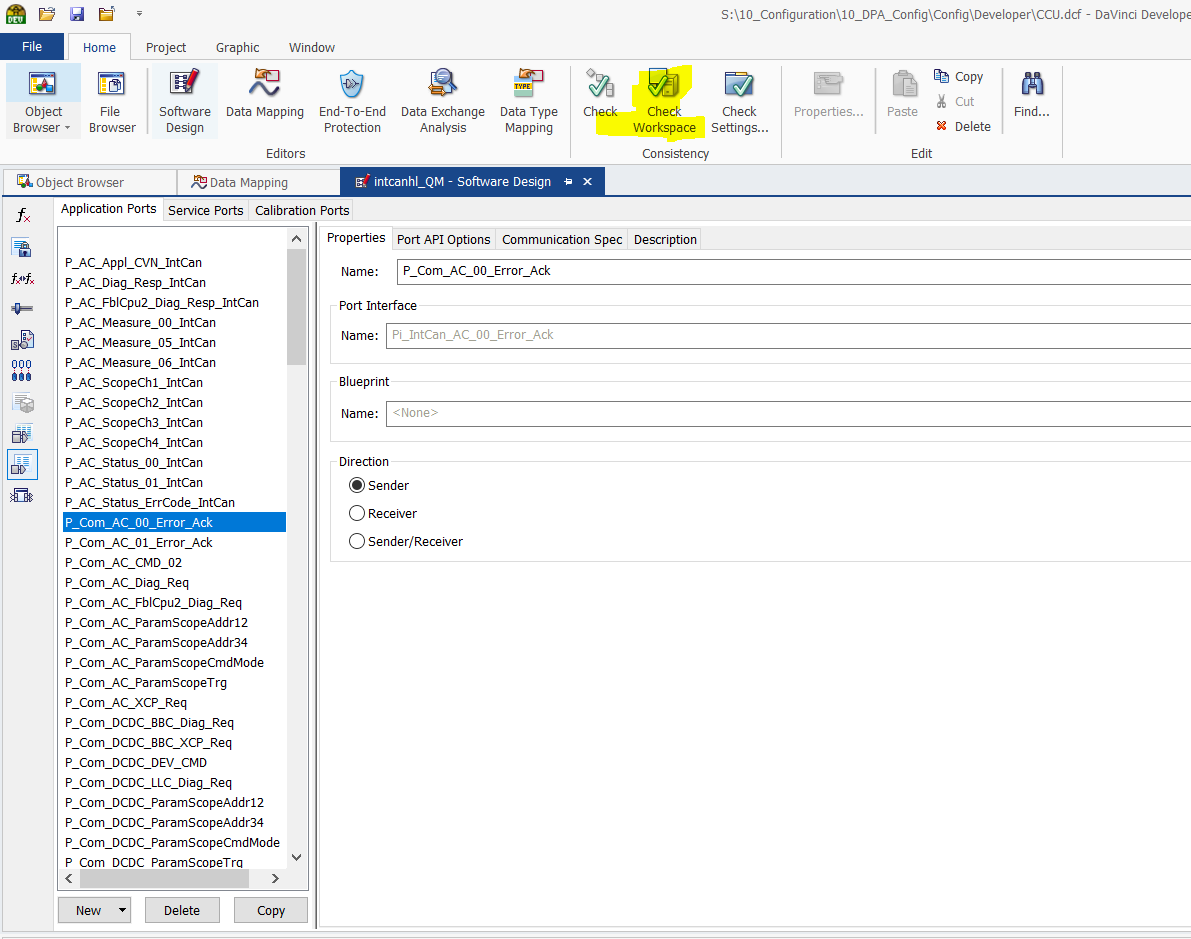


**Step6 :** Do the changes in the existing signals as per the changes in the new dbc file.(like changes in data type , signal length)

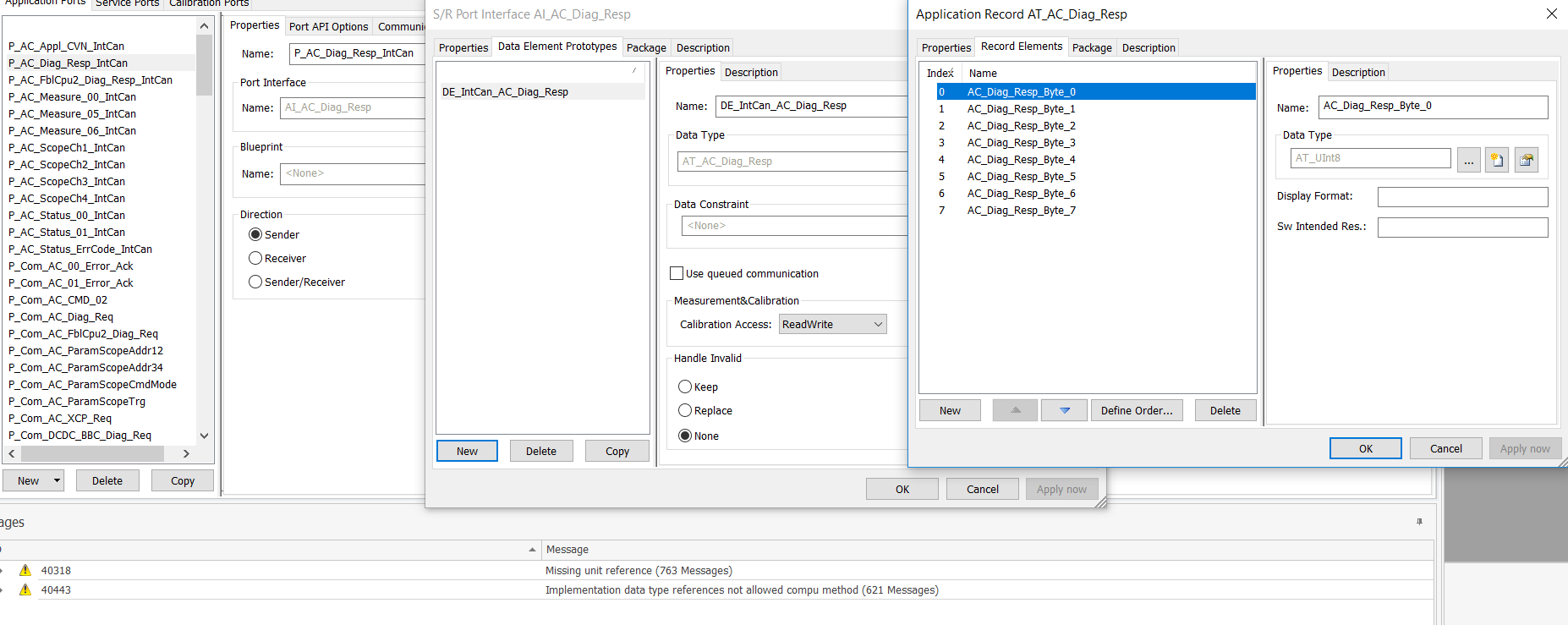
Check for the compu method and Data constraint of the newly created signals



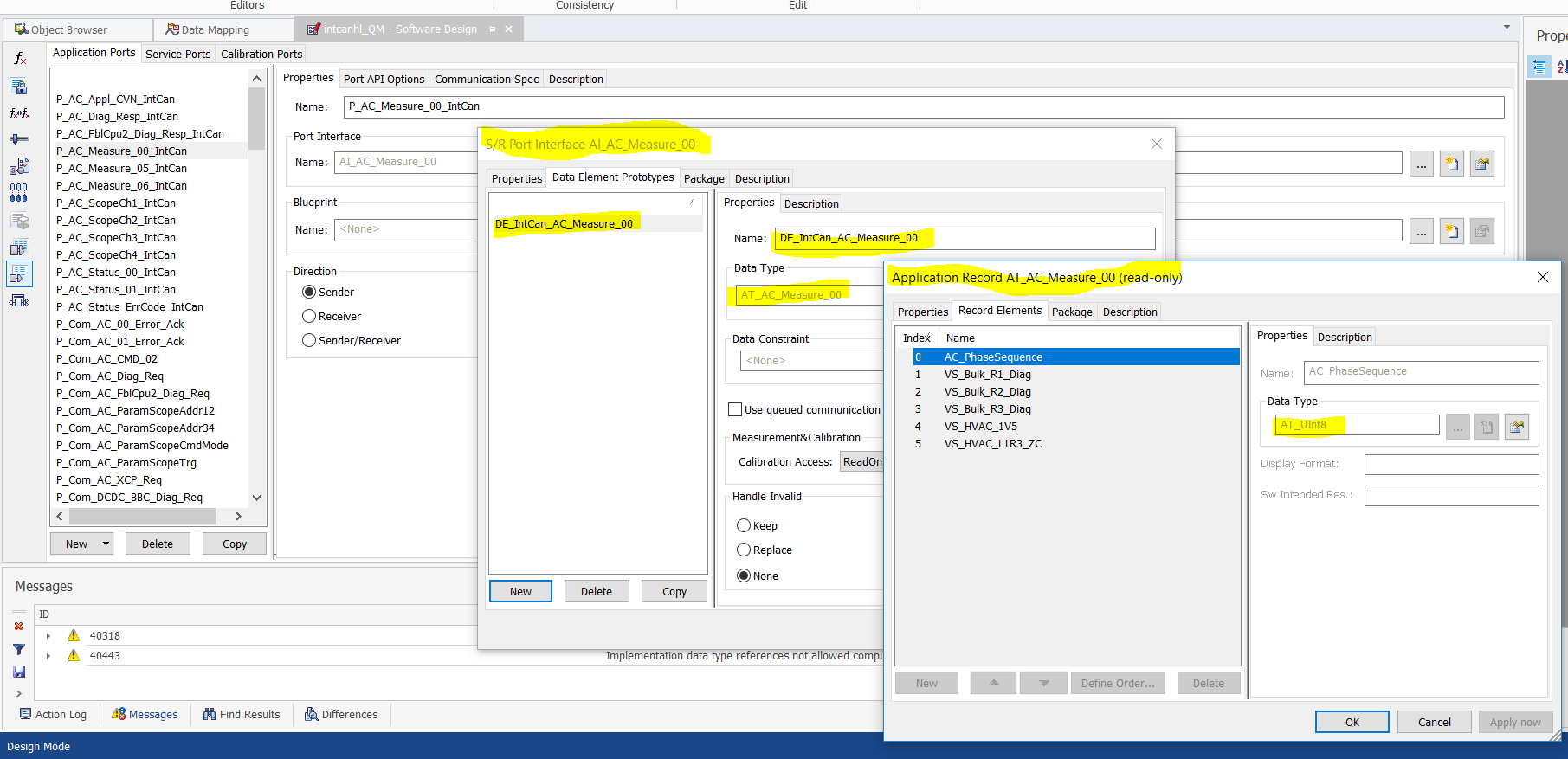
**Step7 :**  Perform workspace check in Davinci developer and resolve the error.



**Step8 :** Take a look at the Application Data type and Implementation Data type



For messages with BZ and CRC should use Qualifier as one of the element



Follow the naming convention as shown below

Pi\_IntCan

Appl Comp

RTE

R\_Com

P\_Com

AI

COM

IntCanhl

Pi\_IntCan : The data type and the Compu method is derived from .dbc file while doing data mappig

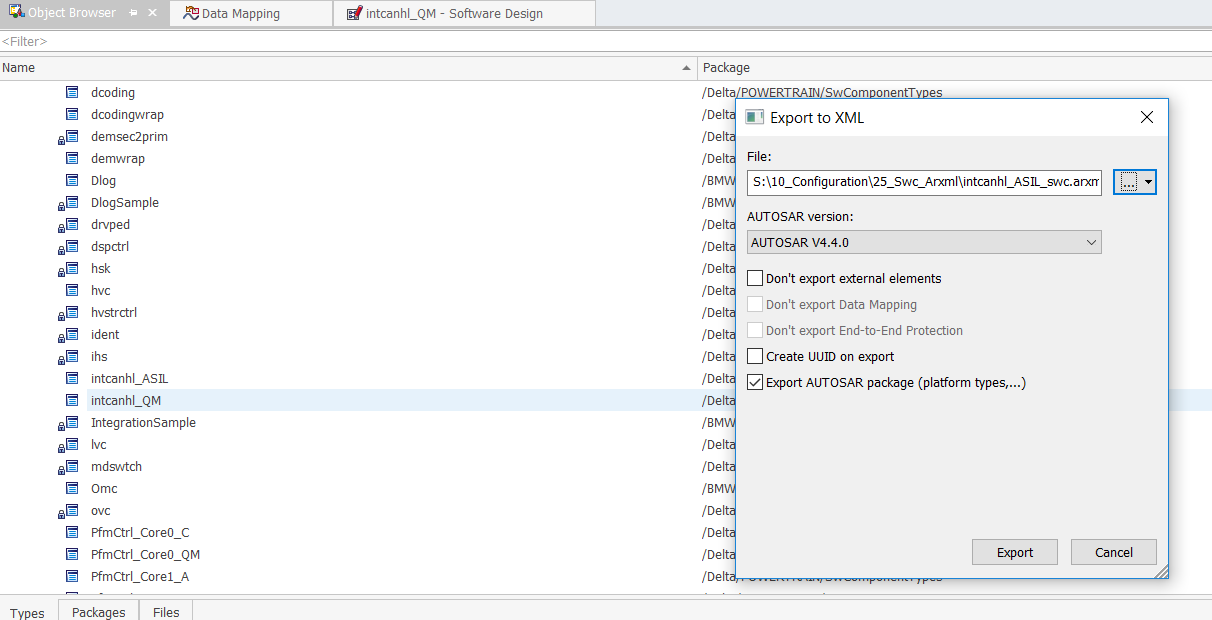
AI : The data type and the Compu method is created manually :Data type is of record type with signals as record elements

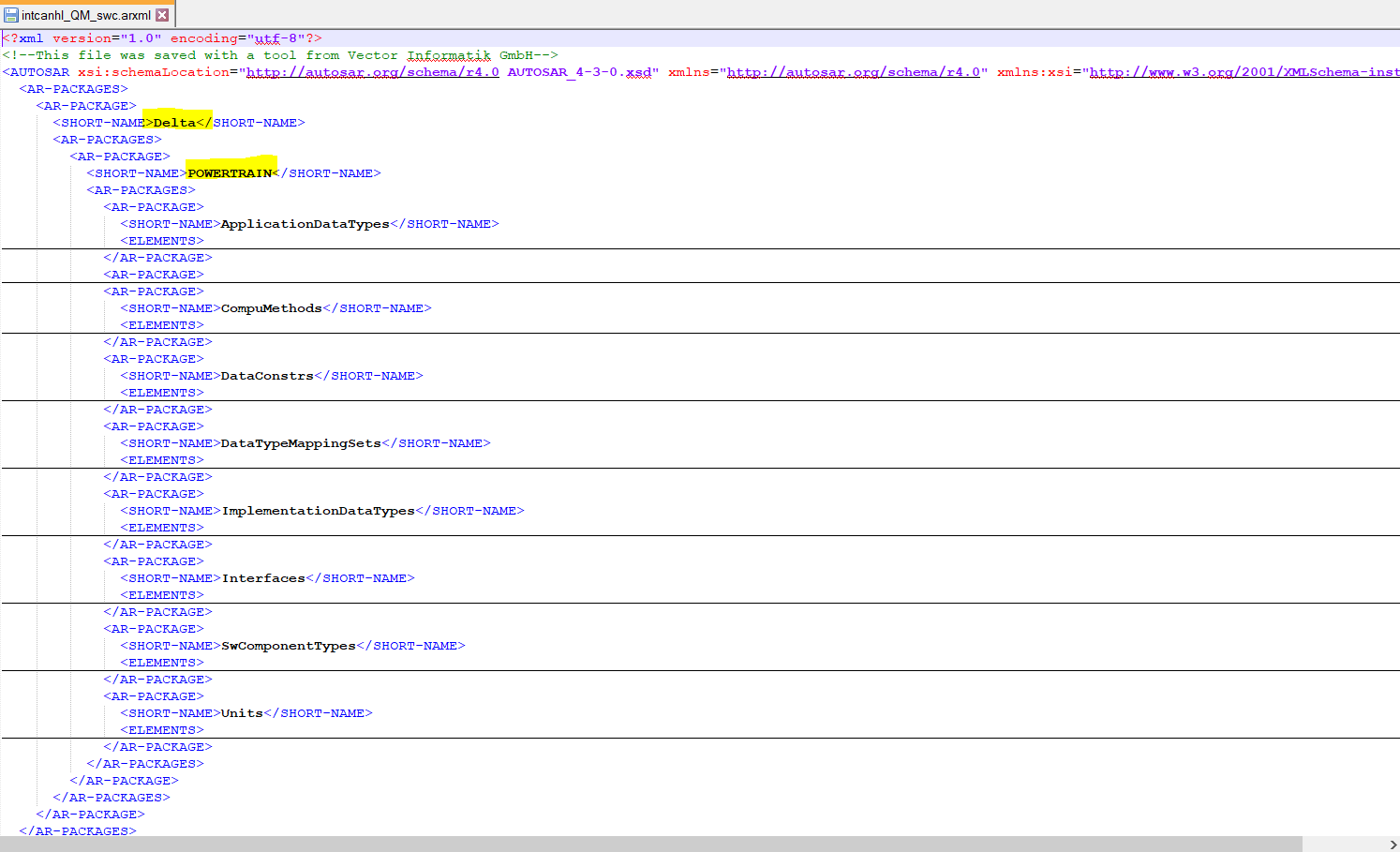
It

Tx runnable : Write port to COM(P\_Com) and the Read port from application(R\_Message\_name\_Intcan)

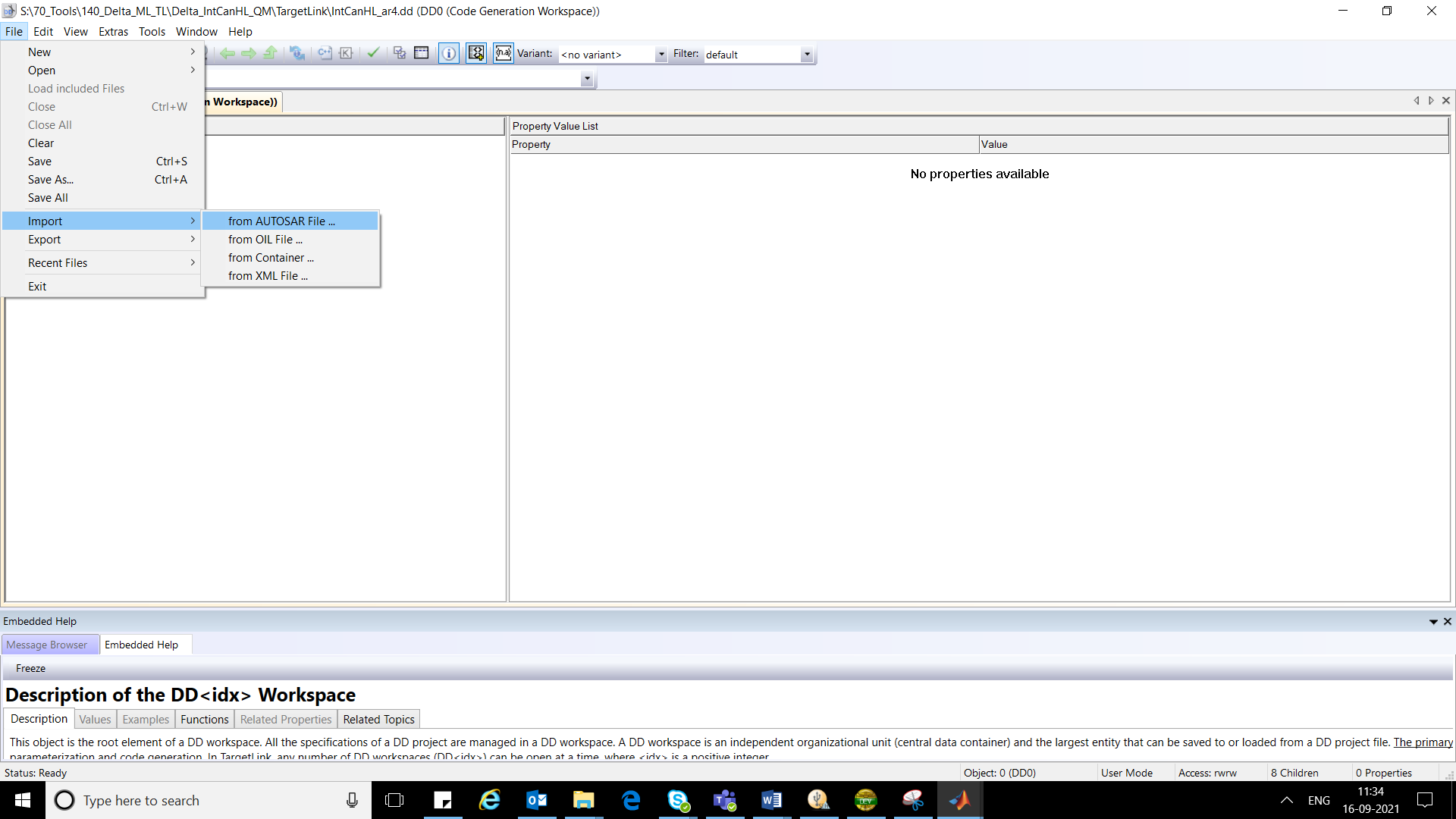
Rx runnable : Read port from COM(R\_Com) (R\_Message\_name) and the Write port to application(P\_mesaage\_name\_Intcan)

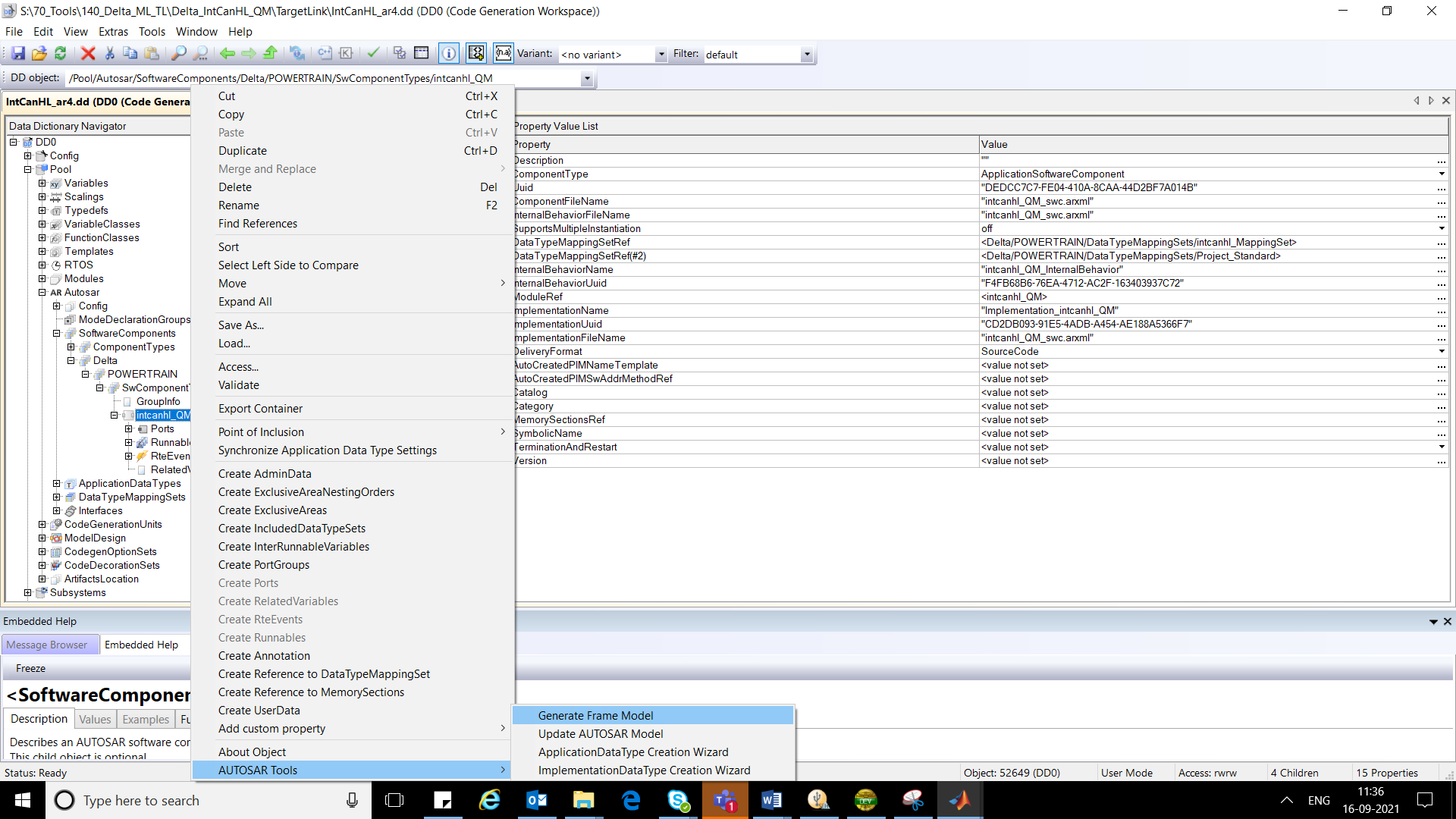
**Step9 :**  Export the SWC in the form of an arxml and check the package in arxml file(it should be delta ->powertrain)





**Step10 :** Import the SWC in targetlink Data dictionary



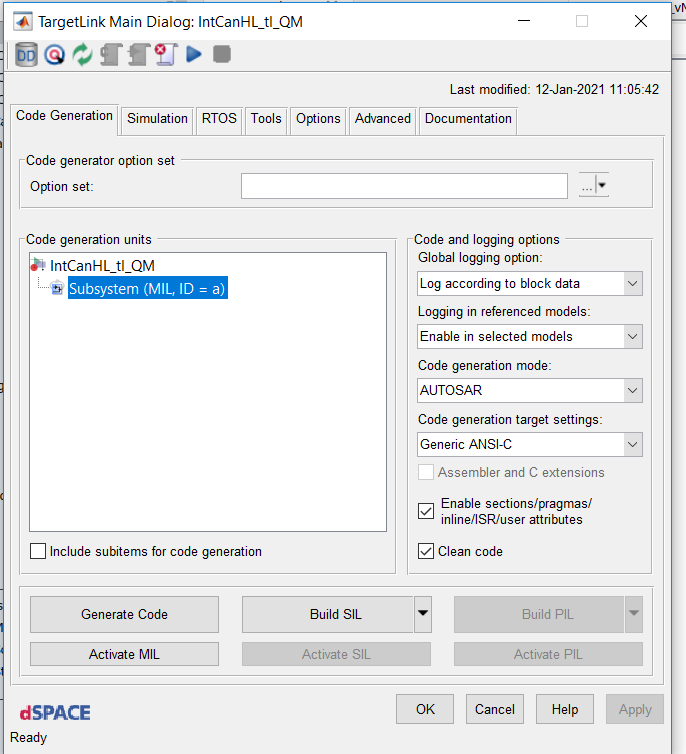
**Step11 :** For the 1st time implementation use the option of generate frame model and after that use the option of update AUTOSAR model.

**Step12 :** Perform the Matlab modeling as per the MAAB guidelines

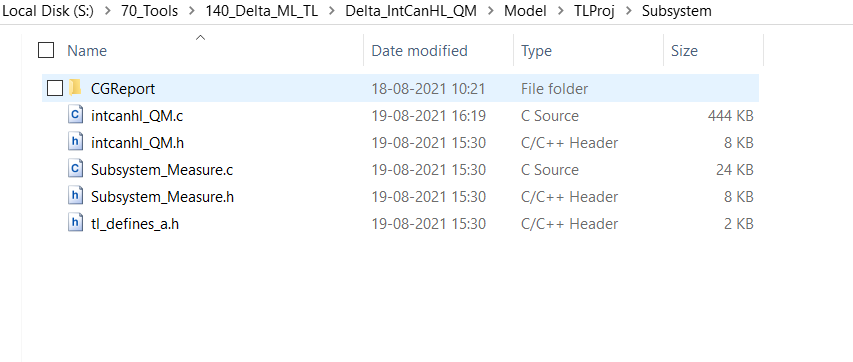
(attached below)

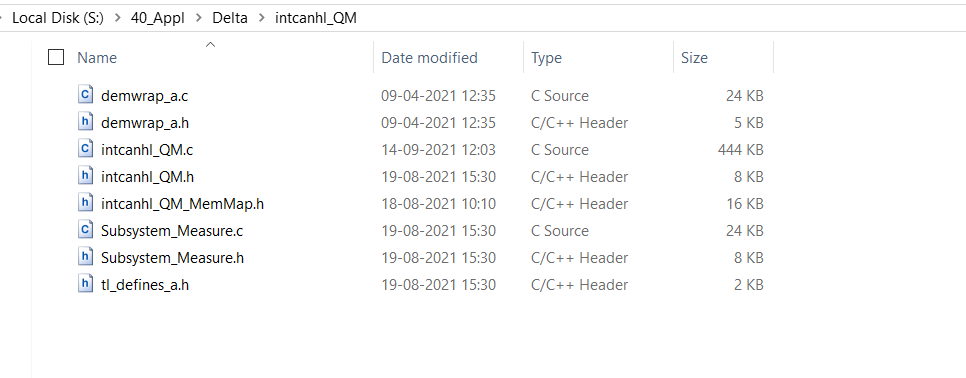


**Step 13:** Generate the code after development.



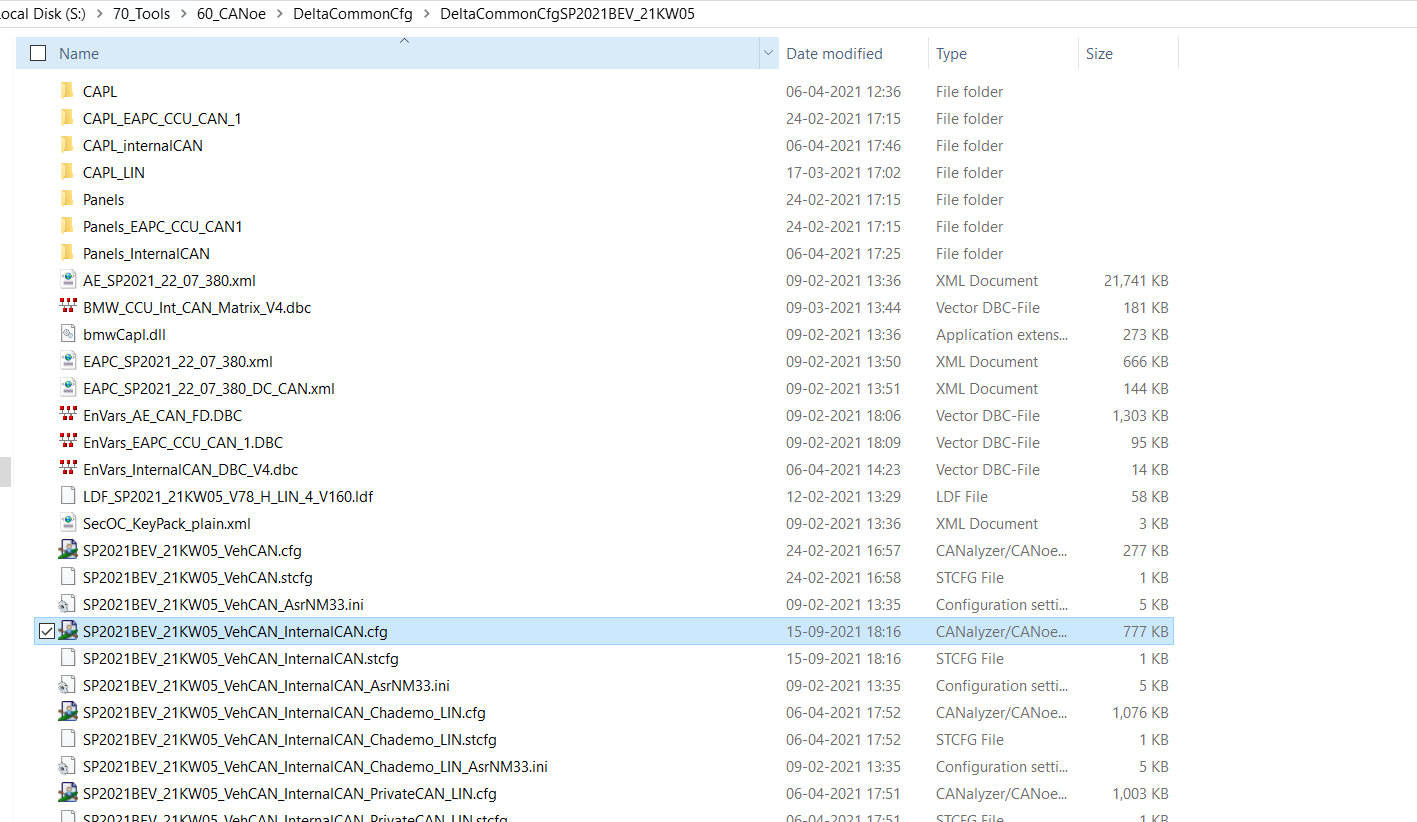
**Step14 :** Copy the generated code in 70\_Tools to 40\_Appl folder



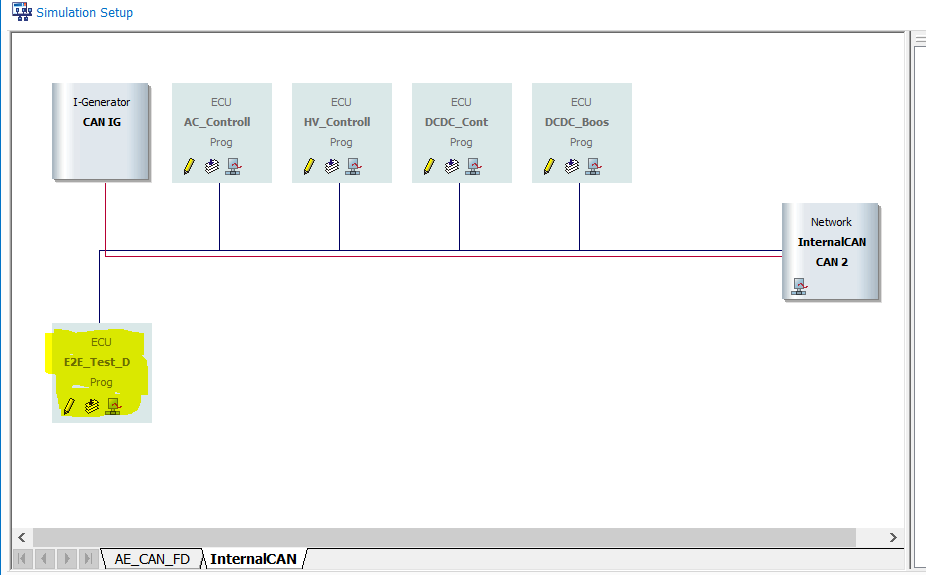


**Step15 :** Compile the code and flash the elf.

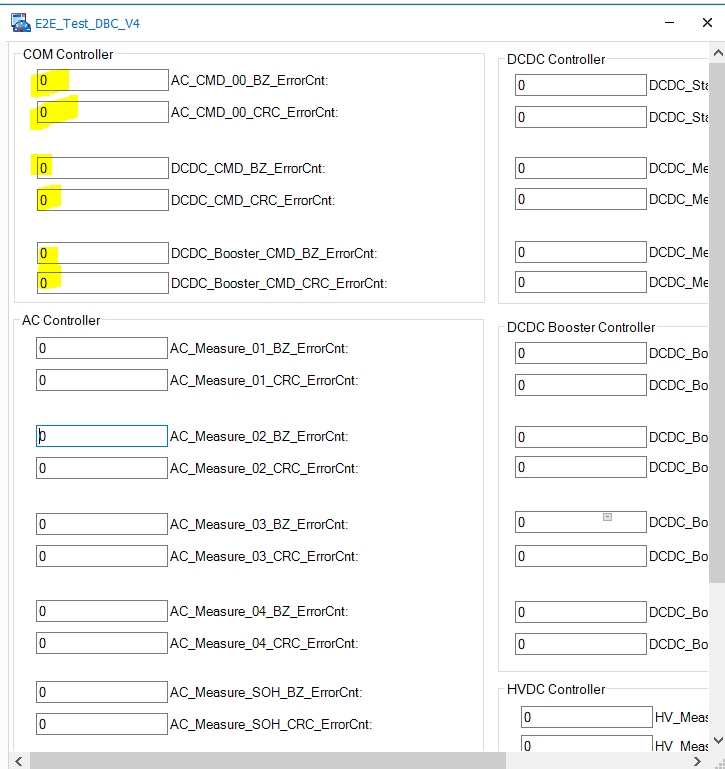
**Step16 :** open Canoe from the below configuration for testing



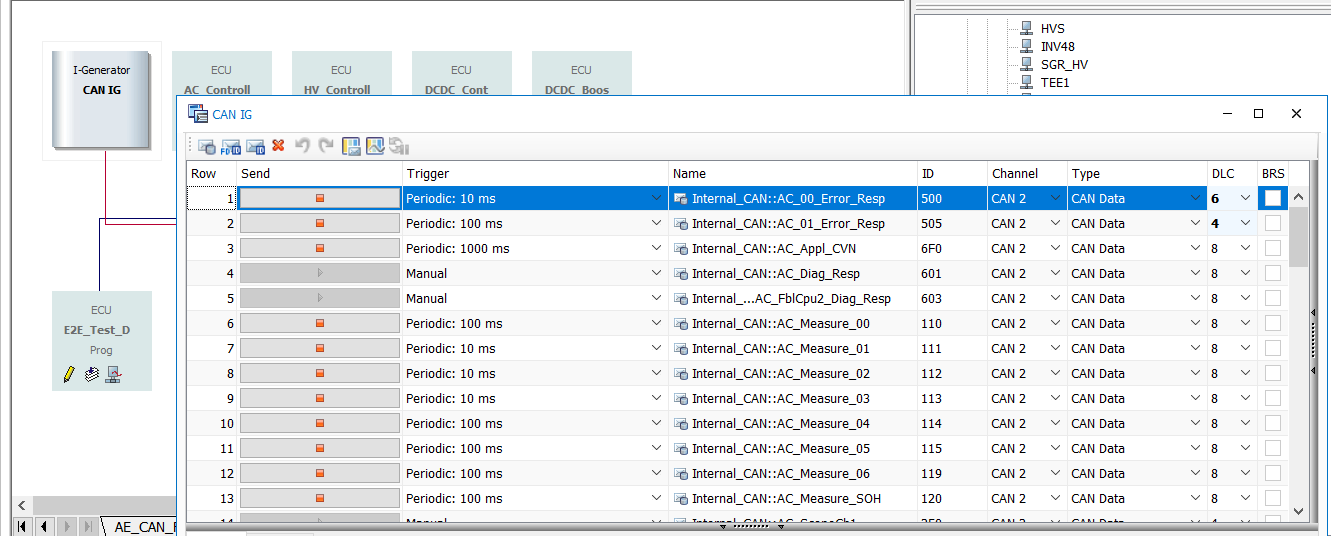
**Step17 :** After opening the canoe enable the IG block and E2E\_Test\_D as shown below to check for the E2E errors in the configuration.



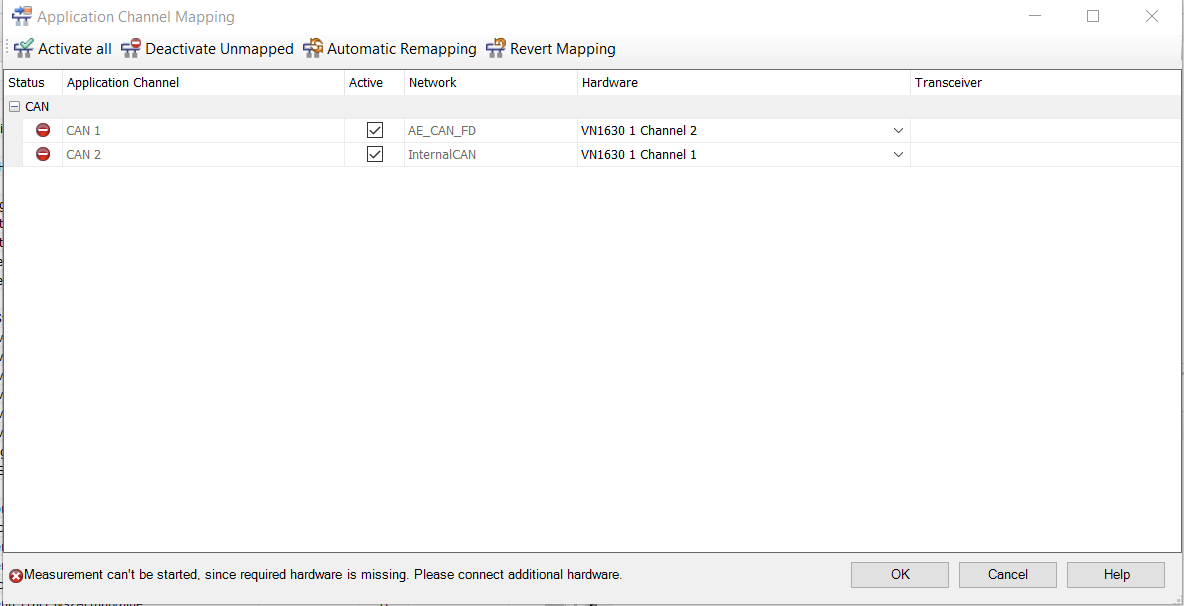
Monitor the Com Controller error counter

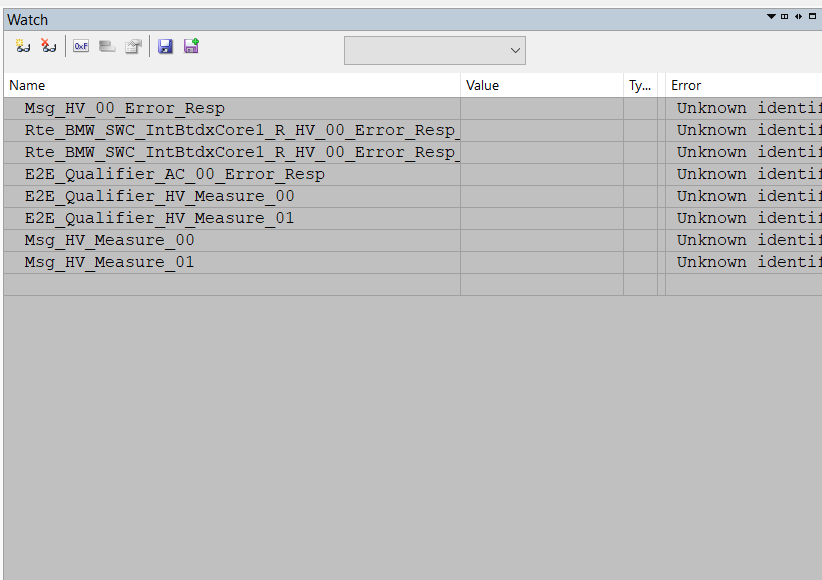


Use the IG block for testing the configuration and transmitting the value to the signals.

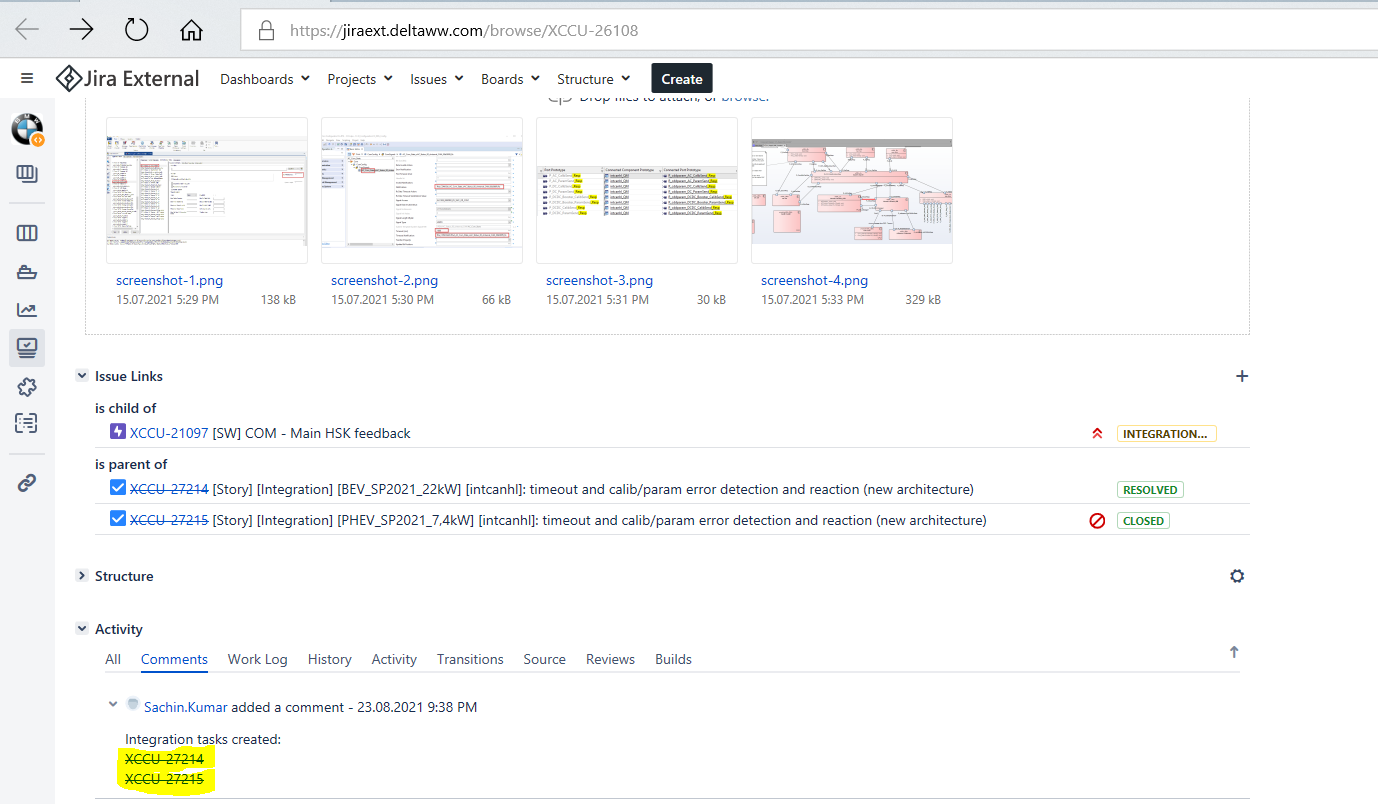


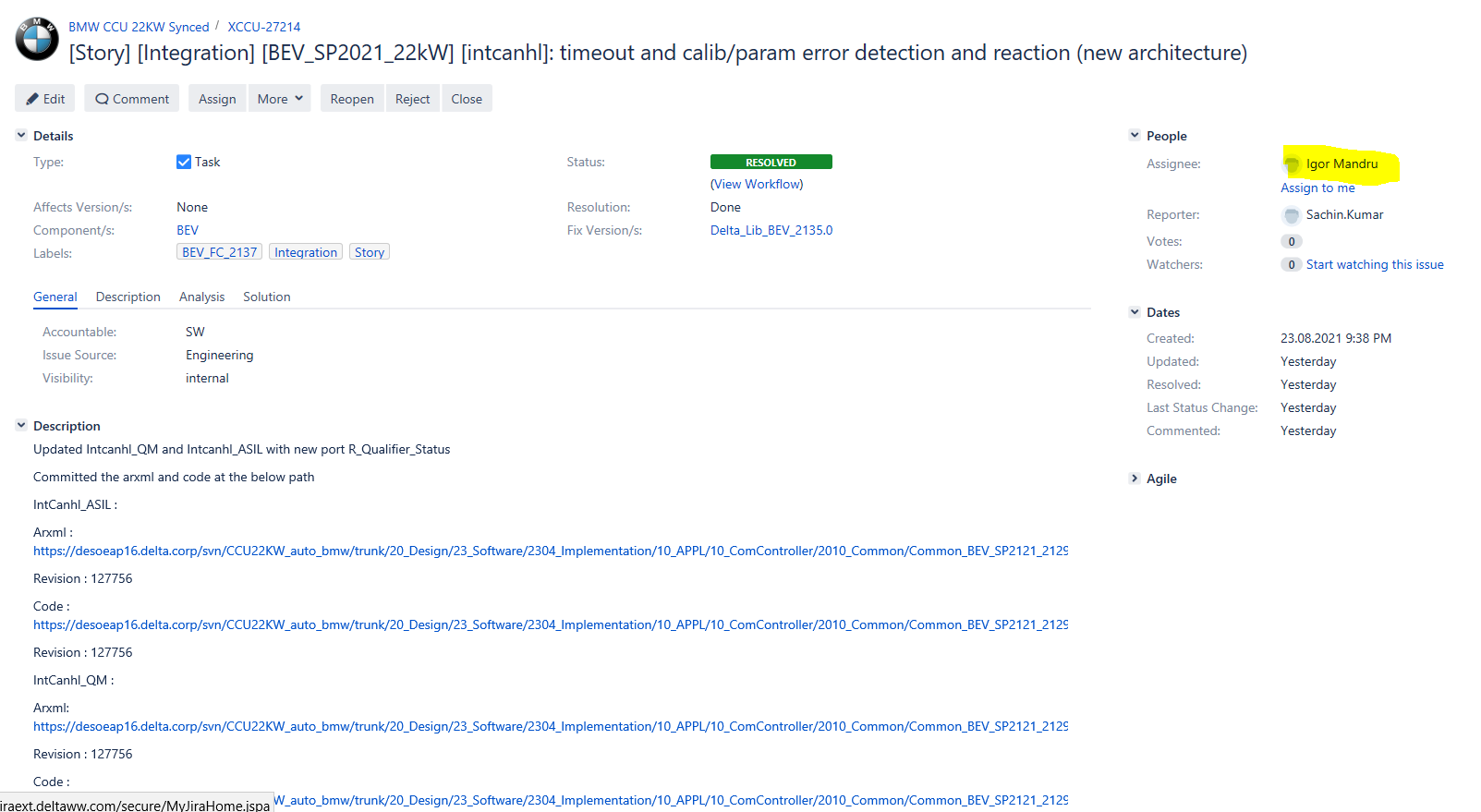
**Step18 :** Provide channel mapping to both Intcan and external CAN from the existing HW channel.External CAN is responsible for keeping the CCU awake.



**Step19 :** Add variable in watch window of Winidea to get the values 

**Step20 :** After testing is completed, commit the code on SVN and create Integration tickets for both BEV and PHEV also assign the tickets to IGOR Mandru





Additional Docs : E2E Spec(attached below)



Document Links CCU

<https://desoeap16.delta.corp/svn/CCU22KW_auto_bmw/trunk/20_Design/23_Software/2300_Management/ProjectHandbook/Delta_SW_Design_Process>

for integration CCU

<https://desoeap16.delta.corp/svn/CCU22KW_auto_bmw/trunk/20_Design/23_Software/2310_HowTo>

for management CCU

<https://desoeap16.delta.corp/svn/CCU22KW_auto_bmw/trunk/20_Design/23_Software/2300_Management>

Document link IPB

<https://desoeap16.delta.corp/svn/IPB_PPE_auto_porsche/trunk/20_Design/23_Software/2303_Software_Architecture_Design>